International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN (P): 2249-6831; ISSN (E): 2249-7943
Vol. 9, Issue 1, Jun 2019, 43-52

TRANS STELLAR
•Journal Publications •Research Consultancy

AN APPLICATION OF ARTIFICIAL INTELLIGENCE WITH VECTOR QUANTIZATION FOR IMAGE COMPRESSION

GIRIDHAR SUDI[1], Dr. MEGHANA KULKARNI[2] & VIKRANT SHENDE[1]

[1]*Assistant Professor, Gogte Institute of Technology, Belagavi, Karnataka, India*

[2]*Associate Professor, Visvesvaraya Technological University, Belagavi, Karnataka, India*

ABSTRACT

This paper uses two compression techniques on an image, namely, Vector Quantization (VQ) and Feed Forward Neural Network (FFNN). VQ is used along with K-Mean clustering to initiate the centroids and form the codebook. The FFNN in this algorithm has an architecture specification of 64 nodes in the input and the output layer along with 16 hidden layers with 16 nodes each. The VQ is applied first on the input image to achieve compression and then the VQ compressed image is fed to the FFNN network for additional compression. A set of observations for compression is recorded for different values of K with a tile size 8. The results are obtained for different values of K such as 50, 100, 150, 200, 250, 500 and 1000. The proposed algorithm gives a compression ratio of about 2 and an acceptable PSNR of about 20db for the standard test image Lena.

Objective

The main objective of this paper is to introduce an algorithm which combines an artificial intelligence technique with a standard compression technique to achieve desirable compression ratios. The flow of this paper is as follows, Section 1 gives an overall introduction about the image compression. Section 2 is about the related work done in image compression where a survey on few related standard papers and their methodologies and results are discussed. Section 3 gives a detailed explanation of the proposed algorithm with flow charts and step wise explanations. Section 4 includes the results and observations obtained through the proposed algorithm. The final section concludes the paper with scope for the future work using this algorithm.

KEYWORDS: Image Compression, Vector Quantization (VQ), K-Mean Clustering, Centroids, Codebook & Feed Forward Neural Network (FFNN)

Received: Apr 13, 2019; **Accepted:** May 03, 2019; **Published:** May 28, 2019; **Paper Id.:** IJCSEITRJUN20196

Original Article

1. INTRODUCTION

Images play an important role in today's digital world, they are used as a representation object. They are widely used in gaming, television, satellites, mobile phones and in medical field. When an image is captured a huge amount of data is also produced which makes it infeasible for storage as well as transmission. A solution for such a problem is image compression, where the original data is reduced by fewer bits without compromising on the image quality, by removing the redundant information and restoring the useful and important information.

There are basically two types of compression techniques:

- Lossless image compression technique.
- Lossy image compression technique.

Lossless compression technique is a form in which compression takes place without loss of any data. It is an exact copy of the original image. Such type of compression has applications in the field of medicine where loss of any data can result in an improper and poor diagnosis, in business documents, text documents, source code etc. On the contrary lossy compression technique is a form in which compression takes place with loss of some redundant and unwanted data, where some compromise on quality is acceptable. Such compression techniques are used where the requirement of compression is high and where some loss of information is acceptable.

1.1 Digital Image Compression

The image compression system basically consists of two parts, one is the compressor and the other being the de-compressor. The compressor includes two parts, viz, pre-processing stage followed by encoding stage, likewise the de-compressor includes the decoding stage then a post-processing stage. The following figures 1 (a) and 1 (b) showa systematic view of the compression process and the de-compressor process.

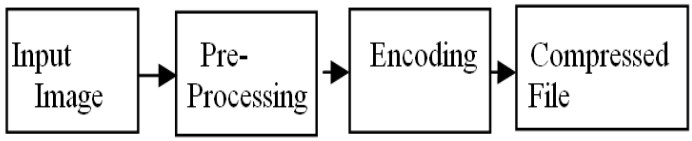

Figure 1(a): Compressor

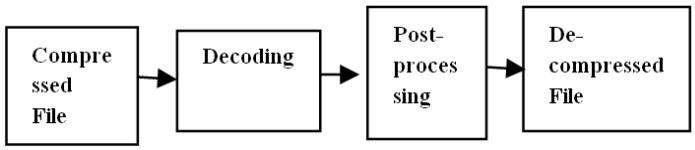

Figure 1(b): De-compressor

The compression system model takes an input image performs pre-processing over the input image. Pre-processing is usually conducted to prepare the image for the next process that is the encoding process. It can consist of operations that are necessary and are application specific. When the compressed file is decoded, the post-processing is performed to eliminate the unwanted and undesirable data gained due to the compression process.

The file with reduced size created by the compression process is called the compressed file. The ratio of the original (uncompressed) file and the compressed file is referred to as the compression ratio. The compression ratio is determined according to the following equation:

$$Compression\ Ratio = \frac{Uncompressed\ file\ size(bytes)}{Compressed\ file\ size\ (bytes)}$$

2. LITERATURE SURVEY

In paper [1] a single hidden layer neural network with four neurons in the hidden layer is used for image compression. A vector quantizer with codebook of 256 code vectors is used in the hidden layer for digital transmission of 0.5 bpp. An input that is a sub- image of size 4x4 pixels, i. e., a total of 16 pixels is given as input to the network. The output vector from the hidden layer which is smaller than the size of the input vector because the input contains 16 neurons whereas the hidden layer consists of only 4 neurons gives the compressed form of the data. The sub- image is reconstructed at output layer which consists of 16 neurons like the input layer. The analysis of results is carried out by

comparing the proposed technique with various other compression techniques that include VQ as residual technique in it. The proposed technique is also compared with 8, 12, 16 hidden neural network. The results show that a good level of PSNR of about 30db is obtained with the proposed technique with different number of neurons in hidden layer than the other compression techniques used in comparison.

Paper [2] is based on image compression using 2- Dimensional Discrete Wavelet Transform (2D-DWT) along with Multistage Vector Quantization (MSVQ). The codebook is generated using LBG algorithm for different stages of vector quantization (VQ). The Radial Basis Function (RBF) neural network is used for training the indices in the MSVQ stages. The method is then applied for different techniques for comparison such as the Discrete Cosine Transform (DCT) and 2-Dimensional Discrete Cosine Transform (2D-DCT). The experiment is applied on six different images of size 128x128 each. The resultant images show that the output obtained from the proposedscheme produces high quality compressed image with better PSNR value and low MSE.

In paper [3] two levels of VQ are applied. One is applied on the transformed image obtained by hybrid wavelet transform and then it is applied on the error image. At both the levels of VQ generation same size of codebook is obtained. At the first level, an input image is compressed using hybrid wavelet transform and a compression ratio of 42.6 is obtained with acceptable image quality but with some blocky effect and some distortion is observed in the image. To get a better and higher compression ratio, VQ is applied on transformed image. The combination of the two techniques, VQ and hybrid wavelet transform increases the compression ratio and eliminates the blocky effect. To reduce the distortion obtained during the hybrid technique, VQ is also applied on error image and both these VQ compressed images are added. Nearly 10% reduction in distortion is obtained.

In paper [4] compression technique is proposed for gray scale medical images using feed forward neural network additionally trained with the back propagation algorithm. The network consists of a three hidden layer feed forward network (FFN) that is applied directly on MRI image as the main compression technique. Training is first performed on sufficient sample images to store the node weight and activation values and then it is applied on the targeted image. The algorithm has compression ratio of 1:30 with PSNR of 39.56 dB. Results indicate that the compression performance such as the PSNR parameters are inversely proportional to the sub- image block size and compression ratio is directly proportional to the number of neurons used in the algorithm.

Paper [5] presents an effective and simple compression coding scheme for the compression of codebook. It precisely makes use of the inter-pixel and inter-block correlation and exploits it effectively to compress the codebook and the indices used in the representation of the code words. In the proposed algorithm the compression of the image takes place in 3 levels, as follows:1: A normal VQ is performed. 2. At the next level compression over the codebook is performed that is the CBC (Codebook Compression) and lastly 3. Search Order Coding (SOC). The algorithm is tried for different sizes of the codebook such as 16, 32, 64, 128, 256,512 and 1024. By uniting these three levels of compression, it can be observed that there is a significant amount of reduction in bit rate of about 0.17 bits per pixel with a decent PSNR value of 29.21 is obtained.

3. METHODOLOGY

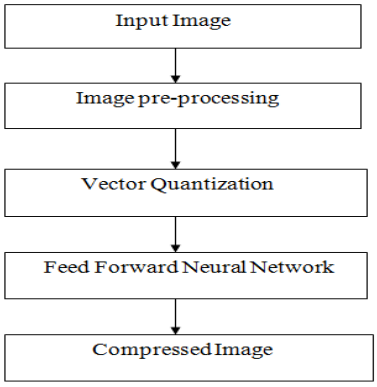

Figure 2: Flowchart for Image Compression

The proposed methodology mainly consists of the 5 steps included in the flowchart as shown in figure 2.

3.1 Input Image

Here an image is read from a file which acts as the original image in the process. This image is fed as an input to the next step and on which the compression takes place. The input image can be of any format such as jpg, png, tiff etc. The file size of the input image is calculated so as to compare it with the final compressed image.

3.2 Image Pre-Processing

Image pre-processing is performed on the input image. It performs some necessary and application specific changes in the input image that makes it ready for the next step of compression. The image pre-processing consists of the steps shown in the figure 3.

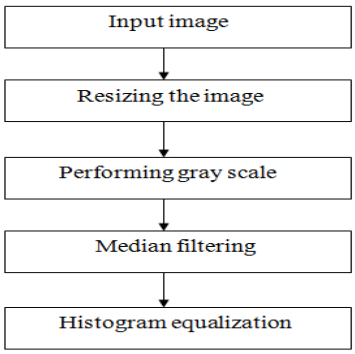

Figure 3: Image Pre-Processing flowchart

3.3 Vector Quantization

Vector Quantization (VQ) is a lossy data compression technique, which is a widely used technology for data storage and transfer. VQ makes use of the rounding off technique or it optimally approximates from an input data to an output data. The compression in VQ is obtained using the 'Codebook', which contains the approximated values using some sort of clustering technique like k-mean clustering. The codebook is used to map the original data or input data to some approximated values which gives the compressed output.

Obtain the image compression we make use of the Vector Quantization along with K-Mean clustering.

The steps are as explained below:

Step 1: Set up a tile size: For example if we set up a tile size of 8, then it picks 8x8=64 pixels at a time to perform the further processing. Here in the present work compressions with different tile sizes have been experimentally observed.

Step 2: Assign a value for K, which represents the number of centroids. These centroids are nothing but tiles. The centroids are the inputs to codebook, therefore the value K are those number of centroids tiles in codebook, which are later used to map the input image for compression. For example if K=200, then there will be 200 tiles in the codebook that will map the remaining tiles based on some classification ordistancemetric.

Step 3: Once the K value is assigned, it randomly assigns k number of centroid tiles in the whole image. For better understanding, let us consider an image of 400x400 pixel resolution, which means it contains 80000 pixels. The tile size is 8, which means it takes 8x8=64 pixels as a single tile. Therefore there are total of 1250 tiles in the image. Now if K=200, then out of 1250 tiles 200 tiles will be randomly selected as the centroid tiles, which are stored in codebook for mapping. The figure 4 below shows an example for k=4, which means there are 4 centroids tiles namely C1, C2, C3 and C4 chosen randomly and these four centroid tiles are stored in the codebook to later map the remaining tiles to obtain compression. Each block in the image below represents a tile.

C3			
		C1	
	C2		
			C3

Figure 4: Centroid Representation in an Image

Step 4: Calculate the mean value of the each centroid. This is done by calculating the mean of each column in centroid tiles and then calculating the mean of those column means. Therefore, we now have mean value for each centroid.

Step 5: This is called the assignment step. Here we randomly assign the remaining tiles of the image to any centroids and this is in the random mode as shown in the figure 5 below.

C3	C2	C1	C3
C2	C3	C1	C4
C1	C2	C3	C2
C2	C4	C1	C3

Figure 5: Assignment of Centroids

Step 6: This is called update step. This step is used to check if the tile randomly assigned to one of the centroids actually belongs to it. We know the centroids have a mean value. In figure 6 the first tile says it belongs to C3. So to check if it really does, the mean of that tile is calculated. Now by using the Euclidean distance the difference between the mean value of the first tile and the mean value of all centroids is calculated. For whichever centroid the mean difference is least the tile is updated with that centroid number as shown in the figure 6. The first tile is updated to C1 and the next tile C3 and so on. It retains the same centroid if after calculating it belongs to the very centroid it was randomly assigned as shown below for the third tile.

~~C3~~ C1	~~C2~~ ~~C3~~	C1	~~C3~~ C1
C2	C3	C1	C4
C1	C2	C3	C2
C2	C4	C1	C3

Figure 6: Update Step of Centroids

Step 7: This step is for the reconstruction of the image. Here the compression of the image takes place using the Codebook. As mentioned previously the codebook contains the K number of tiles which are called centroids with its centroid number. For the reconstruction of the image, it takes the first tile from the above figure, which says it belongs to C1, it now replaces the C1 tile from the Codebook in place of the original tile of the input image. This is how the compression takes place using Vector Quantization along with the K-Mean Clustering.

Step 8: The compression figures obtained through VQ are noted. Now this compressed image is given as an input to the Feed Forward Neural Network for further processing and to obtain further compression.

3.4 Feed Forward Neural Network (FFNN)

A feed forward neural network is considered as a classification algorithm, but here we are using it as a compression technique. The FFNN consists of neurons usually known as the processing units or the nodes. These units or nodes are structured in the form of layers as shown in the figure 7. Every node in a layer is connected with each and every other node in its consecutive layers and each of the node and layer connections contain weight. The data is given to the input layer and it then passes through the network, crossing each layer until it arrives at the outputs and thus they are called feed forward neural networks.

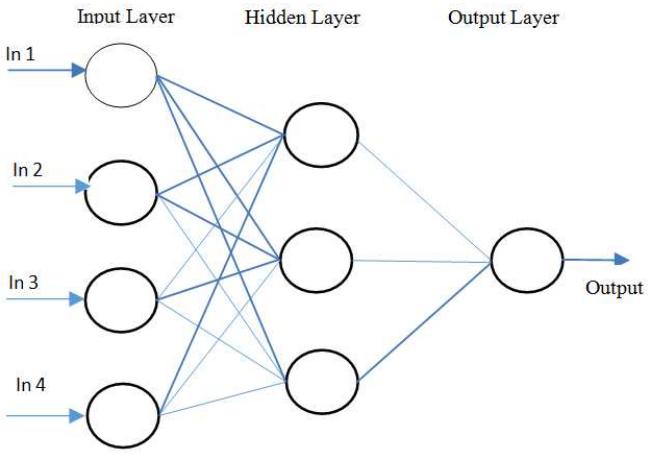

Figure 7: Basic Structure of a Feed Forward Neural Network

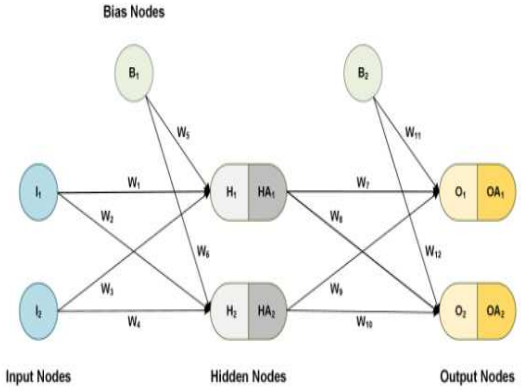

Figure 8: Connection of Nodes and Weights

For the figure 8the specifications are as given below:

W: It represents the weight of the connections. **I**: They are the input nodes or the first layers. **H**: They are the hidden node, which is the weighted sum of input layers or previous hidden layers.

HA: The activated hidden nodes i. e. the value of the hidden node passed to the next nodes.

O: They are the output node or the last layer. **OA**: The activated output node also it is the neural network output.

B: It is known as the Bias node which can be a constant of value typically set to 1.

***e*:** Known as error value which is calculated as the total difference between the output of the network and the desired values.

The steps performed under Feed Forward Neural Network to achieve compression are as explained below:

Step1: This is the initialization step. First we construct the network. We have 64 input and output nodes and 16 hidden layers with 16 nodes each. The structure is such that, each node is connected to every node in its succession layer as shown in figure 8. **Step2:** Assign the weights to the nodes with a random number through normal distribution between 0-1. Set the bias node value to 1.

Step 3: The compressed image produced by the VQ is fed as the input to the neural network to provide further compression. The network fetches the first 8x8=64 pixels as input to the input layer of the network for further processing.

Step 4: Feed-Forward: in this step as the name suggests we calculate the nodes from the input layer to hidden layer and so on. The equation below gives an understanding of the calculation with reference to the figure 8.

$$H_1 = I_1W_1 + I_2W_3 + B_1W_5$$
$$H_2 = I_1W_2 + I_2W_4 + B_1W_6$$

Step 5: Select an activation function for the hidden layer. Here, the Sigmoid function is used as shown below:

$$S(x) = \frac{1}{1+e^{-x}}$$

Calculate hidden node activation values:

$$HA_1 = \frac{1}{1+e^{-H_1}}$$
$$HA_2 = \frac{1}{1+e^{-H_2}}$$

Calculate output node values:

$$O_1 = HA_1W_7 + HA_2W_9 + B_2W_{11}$$
$$O_2 = HA_1W_8 + HA_2W_{10} + B_2W_{12}$$

Calculate output node activation values:

$$OA_1 = O_1$$
$$OA_2 = O_2$$

The final output of the output nodes is placed in an empty matrix for the first 64 pixels. The step 4 is performed continuously until all the tiles are processed in network and replaced in the matrix. After processing all the tiles, a compressed image is formed which has a slight higher compression than the VQ compressed image.

4. RESULTS

Figure 9: Standard Input Image (Lena)

The compression is performed on the standard Lena image as shown in figure 9 above. The input image is of the resolution of 512x512pixel and is of the size 32637 bytes. The compression is performed for different values of K (number of centroids). The range of K depends upon the input image resolution and the tile-size. The tile size chosen is 8, therefore each tile will contain 8x8=64pixels, and the input image resolution is 512x512=262144pixels. No. of tiles = 262144/64= 4096 tiles. Therefore the value of K can range from 0-4096 for 512x512 input image and tile size 8. But, we initially preferred to study the observations for the ideal values of K that range between 50 and 250. Since grey scale image has colour intensities between 0-255. But additionally we can even compare it with higher values than 255. Therefore the values K chosen are 50, 100,150,200,250,500 and 1000.

Table 1: Compression Parameters

	Original Image Size(in kb)	Image Size(In kb) after VQ	Final Image Size after VQ + FFNN (in kb)
K=50	32.637	16.722	15.598
K=100	32.637	18.243	16.661
K=150	32.637	19.701	17.566
K=200	32.637	20.150	17.852
K=250	32.637	21.103	18.534
K=500	32.637	23.492	19.148
K=1000	32.637	25.415	19.975

Table 2: Compressed File Sizes

	Compression Ratio	SNR	PSNR
K=50	2.09	13.67	19.32
K=100	1.95	13.76	19.42
K=150	1.85	13.81	19.47
K=200	1.80	13.85	19.50
K=250	1.70	13.9	19.60
K=500	1.70	13.95	19.61
K=1000	1.63	13.9	19.60

Tables 1 and 2 show the compression parameters and the results obtained. We observe that as the value of K increases from 50 to 1000 the compression ratio decreases and the PSNR and compressed image size increases. We observe that a compression of about half the size of the original image is obtained with an average PSNR of 20db.

5. CONCLUSIONS

Images are an important part of the digital world today. They are used as representation objects in various fields like medicine, satellites, televisions, internet and many more. Therefore, storing and transmitting of these images needs an efficient solution to reduce their cost of storage and transmission. Hence we make use of the various compression techniques. In this paper, a compression algorithm using a combined Vector Quantization(VQ) and Feed Forward Neural Network (FFNN) is introduced. On the input image (standard image Lena) the VQ is applied first using the K-Mean Clustering with a tile size of 8, and considerable compression is achieved. The VQ compressed image acts as an input to FFNN and an additional compression is achieved. The results and observations indicate that an acceptable amount of compression ratio of around 2 which is half of the size of the original image and PSNR of about 20db is achieved. It is observed that as the value of K (number of centroids) increases from 50-1000 for the set of observations, the compression ratio decreases and PSNR increases. Therefore, future work for this algorithm can include working more on the neural network part by introducing the Back Propagation technique into the network. This technique could provide better compression ratios and PSNR's as it keeps iterating in the network with a feedback to the nodes and updating the weights until the desirable result is achieved.

REFERENCES

1. *E. M. Saad, A. A. Abdelwahab and M. A. Deyab, "Using feed forward multilayer neural network and vector quantization as an image data compression technique,"* Computers and Communications, 1998. ISCC '98. Proceedings. Third IEEE Symposium on, *Athens, 1998, pp. 554-558. doi: 10.1109/ISCC.1998.702592*
2. *V. D. Raut and S. Dholay, "Analyzing image compression with efficient transforms & multistage vector quantization using radial basis function neural network,"* 2015 IEEE International Conference on Engineering and Technology (ICETECH),

Coimbatore, 2015, pp.1-6. doi: 10.1109/ICETECH.2015.7275009

3. *P. Natu, S. Natu and T. Sarode, "Hybrid image compression using VQ on error image,"* 2017 International Conference on Intelligent Communication and Computational Techniques (ICCT), *Jaipur, 2017, pp. 173-176. doi: 10.1109/INTELCCT.2017.8324040*
4. *Kumari, V. S. R., & Kumar, P. R. (2013). Cardiac arrhythmia prediction using improved multilayer perceptron neural network. Int. J. Electron., Commun., Instrum. Eng. Res. Develop, 3(4), 73-80.*
5. *W. K. Yeo et al., "Grayscale medical image compression using feedforward neural networks,"* 2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), *Penang, 2011, pp. 633-638. doi: 10.1109/ICCAIE.2011.6162211*
6. *P. K. Shah, R. P. Pandey and R. Kumar, "Vector Quantization with codebook and index compression,"* 2016 International Conference System Modeling & Advancement in Research Trends (SMART), *Moradabad, 2016, pp. 49-52. doi: 10.1109/SYSMART.2016.7894488*
7. *W. Zhang, H. Li and X. Long, "An improved classified vector quantization for medical image,"* 2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA), *Auckland, 2015, pp. 238-241. doi: 10.1109/ICIEA.2015.7334118*
8. *Hussain, A. J., Ali Al-Fayadh, and NaeemRadi. "Image compression techniques: A survey in lossless and lossy algorithms." Neurocomputing (2018).*